

MMM	MMM	AAAAAAAAA	CCCCCCCCCCCCC	RRRRRRRRRRRRR	000000000			
MMM	MMM	AAAAAAAAA	CCCCCCCCCCCCC	RRRRRRRRRRRRR	000000000			
MMM	MMM	AAAAAAAAA	CCCCCCCCCCCCC	RRRRRRRRRRRRR	000000000			
MMMMMMM	MMMMMMM	AAA	AAA	CCC	RRR	RRR	000	000
MMMMMMM	MMMMMMM	AAA	AAA	CCC	RRR	RRR	000	000
MMMMMMM	MMMMMMM	AAA	AAA	CCC	RRR	RRR	000	000
MMM	MMM	AAA	AAA	CCC	RRR	RRR	000	000
MMM	MMM	AAA	AAA	CCC	RRR	RRR	000	000
MMM	MMM	AAA	AAA	CCC	RRR	RRR	000	000
MMM	MMM	AAA	AAA	CCC	RRRRRRRRRRRRR	RRR	000	000
MMM	MMM	AAA	AAA	CCC	RRRRRRRRRRRRR	RRR	000	000
MMM	MMM	AAA	AAA	CCC	RRRRRRRRRRRRR	RRR	000	000
MMM	MMM	AAAAAAAAAAAAAAAAA	CCC	RRR	RRR	RRR	000	000
MMM	MMM	AAAAAAAAAAAAAAAAA	CCC	RRR	RRR	RRR	000	000
MMM	MMM	AAAAAAAAAAAAAAAAA	CCC	RRR	RRR	RRR	000	000
MMM	MMM	AAA	AAA	CCC	RRR	RRR	000	000
MMM	MMM	AAA	AAA	CCC	RRR	RRR	000	000
MMM	MMM	AAA	AAA	CCC	RRR	RRR	000	000
MMM	MMM	AAA	AAA	CCCCCCCCCCCCC	RRR	RRR	000000000	000000000
MMM	MMM	AAA	AAA	CCCCCCCCCCCCC	RRR	RRR	000000000	000000000
MMM	MMM	AAA	AAA	CCCCCCCCCCCCC	RRR	RRR	000000000	000000000

AAAAAA AAAAAA	CCCCCCCC CCCCCCCC	TTTTTTTTTT TTTTTTTTTT	PPPPPPPP PPPPPPPP	RRRRRRRR RRRRRRRR	IIIIII IIIIII	
AA        AA	CC	TT	PP        PP	RR        RR	II	
AA        AA	CC	TT	PP        PP	RR        RR	II	
AA        AA	CC	TT	PP        PP	RR        RR	II	
AA        AA	CC	TT	PPPPPPPP	RRRRRRRR	II	
AA        AA	CC	TT	PPPPPPPP	RRRRRRRR	II	
AAAAAAAAAA	CC	TT	PP	RR    RR	II	
AAAAAAAAAA	CC	TT	PP	RR    RR	II	
AA        AA	CC	TT	PP	RR        RR	II	
AA        AA	CC	TT	PP	RR        RR	II	
AA        AA	CCCCCCCC	TT	PP	RR        RR	IIIIII	....
AA        AA	CCCCCCCC	TT	PP		IIIIII	....

LL	IIIIII	SSSSSSSS
LL	IIIIII	SSSSSSSS
LL	II	SS
LL	II	SS
LL	II	SS
LL	II	SS
LL	II	SSSSSS
LL	II	SSSSSS
LL	II	SS
LL	II	SS
LL	II	SS
LL	II	SS
LLLLLLLLLL	IIIIII	SSSSSSSS
LLLLLLLLLL	IIIIII	SSSSSSSS



(2)	85	DECLARATIONS
(3)	137	PRMUN PRIMARY UNARY OPERATORS
(4)	193	PRMSYM PRIMARY SYMBOLS
(5)	260	NUMERIC PRIMARIES
(6)	317	PROGRAM COUNTER PRIMARY
(7)	351	ENTRY POINT MASK ROUTINES
(8)	420	EXPRESSIONS
(9)	487	UP-ARROW-A ASCII TEXT PRIMARY
(10)	549	RADIX CONTROL
(11)	585	OPERATORS
(12)	632	SYMBOL ATTRIBUTE DIRECTIVES -GLOBL/DEBUG/WEAK/EXTRN

```
0000 1 .TITLE MAC$ACTPRI PRIMARIES
0000 2 .IDENT 'V04-000'
0000 3
0000 4
0000 5 *****
0000 6
0000 7 *
0000 8 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0000 9 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0000 10 * ALL RIGHTS RESERVED.
0000 11 *
0000 12 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0000 13 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0000 14 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0000 15 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0000 16 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0000 17 * TRANSFERRED.
0000 18 *
0000 19 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0000 20 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0000 21 * CORPORATION.
0000 22 *
0000 23 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0000 24 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0000 25 *
0000 26 *****
0000 27
0000 28
0000 29 ++
0000 30 FACILITY: VAX MACRO ASSEMBLER OBJECT LIBRARY
0000 31
0000 32 ABSTRACT:
0000 33
0000 34 The VAX-11 MACRO assembler translates MACRO-32 source code into object
0000 35 modules for input to the VAX-11 LINKER.
0000 36
0000 37 ENVIRONMENT: USER MODE
0000 38
0000 39 AUTHOR: Benn Schreiber, CREATION DATE: 20-AUG-78
0000 40
0000 41 MODIFIED BY:
0000 42
0000 43 V03.01 MTR0013 Mike Rhodes 07-Jun-1982
0000 44 Modify routine EXPBIN to test the Absolute
0000 45 Expression flag MAC$GL_ABSFLAG a little closer
0000 46 in order to interpret the expression type correctly.
0000 47
0000 48 V03.00 MTR0001 Mike Rhodes 15-Mar-1982
0000 49 Modify routine NUMASC to use FLG$V_DLIMSTR flag
0000 50 to allow passing hyphens and semicolons.
0000 51 Fixes SPR #11-42904.
0000 52
0000 53 V02.12 PCG0008 Peter George 28-Aug-1981
0000 54 Fix test for floating negation in PRMUN.
0000 55
0000 56 V02.11 PCG0002 Peter George 05-May-1981
0000 57 Set SYM$M_RELPSECT flag in IDLIST and PRMSYM.
```



0000 58 :  
0000 59 :  
0000 60 :  
0000 61 :  
0000 62 :  
0000 63 :  
0000 64 :  
0000 65 :  
0000 66 :  
0000 67 :  
0000 68 :  
0000 69 :  
0000 70 :  
0000 71 :  
0000 72 :  
0000 73 :  
0000 74 :  
0000 75 :  
0000 76 :  
0000 77 :  
0000 78 :  
0000 79 :  
0000 80 :  
0000 81 :  
0000 82 :  
0000 83 :--

V01.10 RN0023 R. Newland 3-Nov-1979  
New message codes to get error message from system  
message file.

V01.09 RN0014 R. Newland 17-Oct-1979  
Support for G\_floating, H\_floating, and Octaword data types.

V01.07 RN0005 R. Newland 12-Aug-1979  
Remove .ALIGN LONG statements

V01.11 RN0027 R. Newland 14-Jan-1980  
Fix problems with negative floating point literals.  
SPR 11-27884.

V01.08 RN0007 R. Newland 28-Aug-1979  
Fix problem with quadword ^A literals less  
than 8 characters. SPR 11-25674.

V01.05 0003 B. Schreiber 10-JAN-1979  
Catch syntax error if pound sign forgotten before  
ASCII immediate (^A) in operands.

V01.06 0006 B. Schreiber 16-JAN-1979  
Fix problem with data generation if repeated data  
and uparrow-A data (i.e. .BYTE ^A/ /[10])

```
0000 85 .SBTTL DECLARATIONS
0000 86 :
0000 87 : INCLUDE FILES:
0000 88 :
0000 89 :
0000 90 :
0000 91 : MACROS:
0000 92 :
0000 93 :
0000 94 $MAC_SYMBLKDEF ;DEFINE SYMBOL BLOCK OFFSETS
0000 95 $MAC_CTLFLGDEF ;DEFINE CONTROL FLAGS
0000 96 $MAC_INTCODDEF ;DEFINE INT. FILE COMMANDS
0000 97 $MAC_GENVALDEF ;DEFINE OTHER GOOD SYMBOLS
0000 98 $MACMSGDEF ; Define message codes
0000 99 :
0000 100 :
0000 101 : EQUATED SYMBOLS:
0000 102 :
0000 103 :
80000000 0000 104 SIGN_BIT = ^X80000000 ;SIGN BIT
0000 105 :
0000 106 :
0000 107 : OWN STORAGE:
0000 108 :
0000 109 :
00000000 0000 110 .PSECT MAC$RO_DATA,NOEXE,NOWRT,GBL, LONG
0000 111 :
0000 112 :++
0000 113 : THIS DISPATCH TABLE IS USED DURING PASS 1 TO JSB TO
0000 114 : MATH ACTION ROUTINES.
0000 115 :--
0000 116 :
0000 117 P1$ARITH_DISP::
00000000 0000 118 .LONG 0 ;(0)--SHOULD NOT HAPPEN
00000000 0004 119 .LONG P1$ARITH_ADD ;INT$_ADD
00000000 0008 120 .LONG P1$ARITH_AND ;INT$_AND
00000000 000C 121 .LONG P1$ARITH_ASH ;INT$_ASH
00000000 0010 122 .LONG P1$ARITH_DIV ;INT$_DIV
00000000 0014 123 .LONG P1$ARITH_MUL ;INT$_MUL
00000000 0018 124 .LONG P1$ARITH_NEG ;INT$_NEG
00000000 001C 125 .LONG P1$ARITH_NOT ;INT$_NOT
00000000 0020 126 .LONG P1$ARITH_OR ;INT$_OR
00000000 0024 127 .LONG P1$ARITH_SAME ;INT$_SAME
00000000 0028 128 .LONG P1$ARITH_SUB ;INT$_SUB
00000000 002C 129 .LONG P1$ARITH_XOR ;INT$_XOR
0000 0030 130 :
00000000 0000 131 .PSECT MAC$ACTPRI_DATA,NOEXE, LONG
0000 0000 132 :
0000 0000 133 SYM_FLAG:.WORD 0 ;USED FOR GLOBAL/DEBUG/WEAK/EXTERN
0000 0002 134 ENTRY_MASK:
0000 0002 135 .WORD 0 ;USED FOR .ENTRY/.VECTOR
```



```
0004 137 .SBTTL PRMUN PRIMARY UNARY OPERATORS
0004 138
0004 139 :++
0004 140 : FUNCTIONAL DESCRIPTION:
0004 141 :
0004 142 : PRMUN IS CALLED WHEN A UNARY OPERATOR PRODUCTION IS
0004 143 : ENCOUNTERED. IT CHECKS FOR UNARY FLOATING NEGATION
0004 144 : AND CHANGES IT TO AN XOR OF THE SIGN BIT. IF THE
0004 145 : EXPRESSION IS TO BE EVALUATED IN PASS 2 THE INTERMEDIATE
0004 146 : CODE IS EMITTED. THE EXPRESSION IS THEN EVALUATED
0004 147 : AND THE RESULT IS RETURNED IN MAC$GL_VALUE.
0004 148 :
0004 149 : INPUTS:
0004 150 :
0004 151 : MAC$AL_VALSTACK[R7] VALUE OF EXPRESSION
0004 152 : MAC$AL_VALSTACK-4[R7] OPERATION
0004 153 :
0004 154 : OUTPUTS:
0004 155 :
0004 156 : MAC$GL_VALUE COMPUTED VALUE
0004 157 :
0004 158 :--
0004 159
00000000 160 .PSECT MAC$RO_CODE_P1,NOWRT,GBL, LONG
0000 161
0000 162 .ENABL LSB
0000 163
56 0000'CF47 DE 0000 164 PRMUN:: :PRIMITIVE = OPUNARY PRIMITIVE
76 D5 0006 165 MOVAL W^MAC$AL_VALSTACK[R7],R6;POINT TO TOP OF VALUE STACK
000E 166 $VPUSH (R6);PUSH VALUE ONTO STACK
0010 167 TSTL -(R6);BACK UP TO ROUTINE VALUE
0010 168 :
0010 169 : IF OPERATION IS FLOATING NEGATION, CHANGE TO XOR OF SIGN BIT
0010 170 :
06 66 91 0010 171 CMPB (R6),#INT$ _NEG ;ARE WE DOING A NEGATE?
26 12 0013 172 BNEQ 20$ ;IF NEQ NO
0000'CF 91 0015 173 CMPB W^MAC$GB_RDXNDX,- ;YES--AND IS IT A FLOATING NEGATE?
04 0019 174 #RDX$V_FLOAT ;IF LESS NO
1F 1F 001A 175 BLSSU 20$ ;YES--ARE WE EVALUATING ON PASS 2
OC 6B 06 E1 001C 176 BBC #FLG$V_EVALEXPR,(R11),10$ ;Yes--stack floating point sign bit
66 0B 9A 002C 177 $INTOUT LW INT$ STKL,<#^X8000> ;CHANGE COMMAND TO XOR
002F 179 MOVZBL #INT$ XOR,(R6) ;Stack sign bit
50 66 D0 003B 180 20$: MOVL (R6),R0 ;GET ACTION
56 50 D0 003E 181 MOVL R0,R6 ;REMEMBER IT FOR LATER
OB 6B 06 E1 0041 182 BBC #FLG$V_EVALEXPR,(R11),30$ ;BRANCH IF NOT EVAL ON PASS 2
51 02 9A 0045 183 MOVZBL #2,R1 ;SET BYTE COUNT
FFB5' 30 0048 184 BSBW MAC$INTOUT X ;EMIT TO INT. FILE
0000'CF 59 D0 004B 185 MOVL R9,W^MAC$GL_EXPEND ;SAVE END OF EXPR POINTER
56 0000'CF46 D0 0050 186 30$: MOVL W^P1$ARITH_DISP[R6],R6 ;GET ADDRESS OF ROUTINE
66 16 0056 187 JSB (R6) ;CALL IT
0058 188 $VPOP W^MAC$GL_VALUE ;RETRIEVE VALUE
05 0062 189 RSB
0063 190
0063 191 .DSABL LSB
```

```
0063 193 .SBTTL PRMSYM PRIMARY SYMBOLS
0063 194
0063 195 :++
0063 196 : FUNCTIONAL DESCRIPTION
0063 197 :
0063 198 : PRMSYM IS INVOKED WHEN AN ID IS FOUND IN THE PRODUCTION.
0063 199 : BASED ON THE SYMBOL ATTRIBUTES (LOCAL, GLOBAL, EXTERNAL,
0063 200 : DEFINED, ABSOLUTE) IT WILL SET CONTROL FLAGS FOR LATER
0063 201 : PROCESSING OF THE ID.
0063 202 :
0063 203 : INPUTS:
0063 204 :
0063 205 : MAC$GL_VALUE POINTER TO ID SYMBOL BLOCK
0063 206 :
0063 207 : OUTPUTS:
0063 208 :
0063 209 :--
0063 210
0063 211 PRMSYM:: :PRIMARY = ID
0063 212
0063 213 MOVL W^MAC$GL_VALUE,R6 :GET POINTER TO SYMBOL BLOCK
0063 214 BBS #FLG$V_NOREF,(R11),5$ :BRANCH IF WE SHOULD NOT REF SYMBOL
0063 215 BISW2 #SYM$M_REF,SYM$W_FLAG(R6) :FLAG SYMBOL AS REFERENCED
0063 216 MOVL MAC$GL_PSECTPTR,R0 :GET POINTER TO PSECT DATA
0063 217 BBC #PSC$V_REL,- :IF ABS PSECT
0063 218 PSC$W_OPTIONS(R0),5$ :THEN SKIP
0063 219 BISW2 #SYM$M_RELPSECT,SYM$W_FLAG(R6) :SET REL PSECT FLAG
0063 220 5$: BICW2 #SYM$M_SUPR,SYM$W_FLAG(R6) :AND CLEAR SUPPRESS BIT
0063 221 BBS #SYM$V_DEF,SYM$W_FLAG(R6),10$
0063 222 :IF SYMBOL NOT YET DEFINED
0063 223 BICL2 #FLG$M_COMPEXPR,(R11) :THEN EXPR VALUE NOT YET KNOWN
0063 224 10$: BITW #SYM$M_GLOBL!SYM$M_EXTRN,- :SYMBOL GLOBAL OR EXTERNAL?
0063 225 SYM$W_FLAG(R6)
0063 226 BEQL 20$ :IF EQL NO
0063 227 BITW #SYM$M_DEF!SYM$M_LOCAL,- :YES--DEFINED OR LOCAL?
0063 228 SYM$W_FLAG(R6)
0063 229 BNEQ 20$ :IF NEQ NO
0063 230 :
0063 231 :SYMBOL IS EXTERNAL OR GLOBAL
0063 232 :SYMBOL NOT YET DEFINED
0063 233 BBC #FLG$V_EVALEXPR,(R11),50$ :EVALUATE ON PASS 2?
0063 234 $INTOUT_LW INT$_STKG,R6 :YES--STACK GLOBAL
0063 235 MOVL -R9,W^MAC$GL_EXPEND :SAVE END OF EXPRESSION
0063 236 BRB 50$
0063 237 :
0063 238 : LOCAL OR DEFINE SYMBOL
0063 239 :
0063 240 20$: CMPB SYM$B_SEG(R6),W^MAC$GL_PRMSEG :DIFFERENT PSECTS?
0063 241 BEQL 30$ :IF EQL NO
0063 242 BBS #SYM$V_ABS,SYM$W_FLAG(R6),30$ :YES--UNLESS SYMBOL ABSOLUTE
0063 243 : (BRANCH IF ABSOLUTE)
0063 244 TSTL W^MAC$GL_PRMSEG :REALLY DIFFERENT PSECTS?
0063 245 BEQL 30$ :IF EQL NO
0063 246 BICL2 #FLG$M_COMPEXPR,(R11) :YES--VALUE NOT YET KNOWN
0063 247 30$: BBC #FLG$V_EVALEXPR,(R11),40$ :EVALUATE ON PASS 2?
0063 248 $INTOUT_LW INT$_STKS,R6 :YES--STACK SYMBOL
0063 249 MOVL -R9,W^MAC$GL_EXPEND :SAVE END OF EXPRESSION
```

56 0000'CF D0 0063 213  
18 6B 18 E0 0068 214  
09 A6 0080 8F A8 006C 215  
50 00000000'EF D0 0072 216  
06 0D A0 03 E1 0079 217  
09 A6 0800 8F A8 007E 218  
09 A6 4000 8F AA 0084 219  
03 09 A6 00 E0 008A 220  
6B 04 CA 008F 221  
0C B3 0092 222  
09 A6 0094 223  
1B 13 0096 224  
0041 8F B3 0098 225  
09 A6 009C 226  
13 12 009E 227  
00A0 228  
00A0 229  
00A0 230  
3C 6B 06 E1 00A0 231  
0000'CF 59 D0 00A4 232  
2D 11 00AC 233  
00B1 234  
00B3 235  
00B3 236  
00B3 237  
0000'CF 0C A6 91 00B3 238  
0E 13 00B9 239  
09 09 A6 04 E0 00BB 240  
0000'CF D5 00C0 241  
03 13 00C4 242  
6B 04 CA 00C6 243  
0D 6B 06 E1 00C9 244  
00CD 245  
0000'CF 59 D0 00D5 246



HEX	ASSEMBLY	ADDRESS	DISASSEMBLY
0000'CF 05 A6 D0	00DA	250	40\$: MOVL SYMSL VAL(R6),W^MAC\$GL_VALUE ;VALUE IS VALUE OF SYMBOL
04 09 A6 04 E1	00E0	251	50\$: BBC #SYMSV_ABS,SYMSW_FLAG(R6),60\$ ;IS SYMBOL ABSOLUTE?
10 6B 02 E0	00E5	252	BBS #FLGSV_COMPEXPR,(R11),70\$ ;YES--DO WE KNOW EXPR VALUE?
0000'CF D6	00E9	253	60\$: INCL W^MAC\$GL_ABSFLAG ;NO--NOT ABSOLUTE EXPRESSION
0000'CF D5	00ED	254	TSTL W^MAC\$GL_PRMSEG ;DOES EXPR HAVE A SEG YET?
06 12 00F1		255	BNEQ 70\$ ;IF NEQ YES
0000'CF 0C A6 9A	00F3	256	MOVZBL SYMSB_SEG(R6),W^MAC\$GL_PRMSEG ;NO--USE SYMBOL SEGMENT
55 00'8F 9A	00F9	257	70\$: MOVZBL #CRFSR_REF,R5 ;SET REFERENCE
FF00' 31	00FD	258	BRW MAC\$CREF_SYM ;CREF SYMBOL IF CREFFING AND RETURN

```
.SBTTL NUMERIC PRIMARIES

0100 260
0100 261
0100 262 :++
0100 263 : FUNCTIONAL DESCRIPTION:
0100 264 :
0100 265 : NUMFLT IS CALLED WHEN '^F' IS SEEN. A FLOATING POINT NUMBER
0100 266 : IS SCANNED.
0100 267 :
0100 268 :--
0100 269
0100 270 NUMFLT:: :SPECIAL OPERATOR = DUPF
FEFD' 30 0100 271 BSBW MAC$SKIPSP :SKIP SPACES
FEFA' 30 0103 272 BSBW MAC$GETFLOAT :ACCUMULATE FLOATING POINT NUMBER
00 11 0106 273 BRB PRMINT :TREAT AS INTEGER
0108 274
0108 275 :++
0108 276 : FUNCTIONAL DESCRIPTION:
0108 277 :
0108 278 : PRMINT IS CALLED WHEN AN INTEGER (OR INTEGER-LIKE) TOKEN
0108 279 : IS FOUND. IF THE EXPRESSION IS BEING EVALUATED IN PASS 2
0108 280 : THE VALUE IS EMITTED TO THE INTERMEDIATE FILE.
0108 281 :
0108 282 :--
0108 283
0108 284 PRMINT:: :PRIMARY = DINTEGER
OF 6B 06 E1 0108 285 BBC #FLGSV EVALEXPR,(R11),10$ :EVALUATE ON PASS 2?
0000'CF 59 D0 010C 286 $INTOUT_LW INT$ STKL,<W^MAC$GL_VALUE> :YES--STACK VALUE
05 0116 287 MOVL -R9,W^MAC$GL_EXPEND :SAVE END OF EXPRESSION
011B 288 10$: RSB
011C 289
011C 290 :++
011C 291 : FUNCTIONAL DESCRIPTION:
011C 292 :
011C 293 : PRMBRK IS CALLED WHEN AN EXPRESSION IN ANGLE BRACKETS IS
011C 294 : SCANNED. THE VALUE IS PICKED OFF OF THE STACK AND PLACED
011C 295 : IN MAC$GL_VALUE.
011C 296 :
011C 297 :--
011C 298
011C 299 PRMBRK:: :PRIMARY = DANGOPN EXPR DANGCLS
0000'CF FFFC'CF47 D0 011C 300 MOVL W^MAC$AL_VALSTACK-4[R7],- :VALUE IS ON STACK
0124 301 W^MAC$GL_VALUE ;
05 0124 302 RSB
0125 303
0125 304 :++
0125 305 : FUNCTIONAL DESCRIPTION:
0125 306 :
0125 307 : PRMRDX IS CALLED WHEN A RADIX CONTROL PRIMARY HAS BEEN
0125 308 : SCANNED. THE RADIX IS RESET TO THE PREVIOUS RADIX.
0125 309 :
0125 310 :--
0125 311
0125 312 PRMRDX:: :PRIMARY = RADIX CONTROL PRIMARY
0000'CF FFFC'CF47 F6 0125 313 CVTLB W^MAC$AL_VALSTACK-4[R7],- :RESET TO PREVIOUS
012D 314 W^MAC$GB_RDXNDX :RADIX
05 012D 315 RSB
```



```
012E 317 .SBTTL PROGRAM COUNTER PRIMARY
012E 318
012E 319 :++
012E 320 : FUNCTIONAL DESCRIPTION:
012E 321
012E 322 PRMPC IS CALLED WHEN A PC REFERENCE ('.') IS SCANNED.
012E 323 IF THE EXPRESSION IS BEING EVALUATED ON PASS 2 THE
012E 324 CODE IS EMITTED TO STACK THE PC VALUE. IF THE EXPRESSION
012E 325 CONTAINS CROSS-PSECT REFERENCES THEN THE EXPRESSION IS
012E 326 NOT A COMPILE-TIME EXPRESSION AND THE FLAG (FLG$M_COMPEXPR)
012E 327 IS CLEARED IN THE FLAGS WORD.
012E 328
012E 329 :--
012E 330
012E 331 PRMPC::
012E 332 BBC #FLG$V_EVAEXPR,(R11),10$ :PRIMARY = DPC
0132 333 $INTOUT_X INT$ STKPC :BR IF NOT EVALUATE ON PASS 2
0138 334 MOVL R9,W^MAC$GL_EXPEND :YES--STACK PC
013D 335 10$: MOVL W^MAC$GL_PC,W^MAC$GL_VALUE :SAVE END OF EXPRESSION
0144 336 MOVL W^MAC$GL_PSECTPTR,R0 :RETURN VALUE IS PC
0149 337 BBC #PSC$V_REL,PSC$W_OPTIONS(R0),30$ :GET POINTER TO CURRENT PSECT
014E 338 :BRANCH IF ABS PSECT
014E 339 INCL W^MAC$GL_ABSFLAG :RELOCATABLE--FLAG NOT ABS EXPR
0152 340 TSTL W^MAC$GL_PRMSEG :EXPR HAVE A PSECT YET?
0156 341 BNEQ 20$ :IF NEQ YES
0158 342 MOVL W^MAC$GL_PSECT,- :NO--USE CURRENT PSECT
015C 343 W^MAC$GL_PRMSEG
015F 344 BRB 30$
0161 345 20$: CMPL W^MAC$GL_PRMSEG,- :YES--CROSS PSECT REFERENCES?
0165 346 W^MAC$GL_PSECT
0168 347 BEQL 30$ :IF EQL NO
016A 348 BICL2 #FLG$M_COMPEXPR,(R11) :YES--FLAG NOT COMPILE EXPRESSION
016D 349 30$: RSB
```

OB 6B 06	E1	012E	332
0000'CF 59	D0	0132	333
0000'CF 0000'CF	D0	0138	334
50 0000'CF	D0	013D	335
1F 0D A0 03	E1	0144	336
		0149	337
		014E	338
0000'CF D6		014E	339
0000'CF D5		0152	340
09 12		0156	341
0000'CF D0		0158	342
0000'CF		015C	343
0C 11		015F	344
0000'CF D1		0161	345
0000'CF		0165	346
03 13		0168	347
6B 04	CA	016A	348
	05	016D	349

```
016E 351 .SBTTL ENTRY POINT MASK ROUTINES
016E 352
016E 353 :++
016E 354 : FUNCTIONAL DESCRIPTION:
016E 355 :
016E 356 : RGLST1 AND REGLST ARE CALLED TO ACCUMULATE AN ENTRY-POINT
016E 357 : MASK. RGLST1 IS CALLED FOR THE FIRST ITEM TO INITIALIZE THE
016E 358 : ENTRY MASK TO ZERO, AND REGLST IS CALLED FOR EACH SUCCESSIVE
016E 359 : ITEM IN THE MASK. THE APPROPRIATE BIT IN ENTRY_MASK IS
016E 360 : SET FOR LATER PROCESSING BY THE 'MASK' ROUTINE.
016E 361 :
016E 362 :--
016E 363
016E 364 RGLST1:: :REGLIS = MASK_ITEM
0002'CF B4 016E 365 CLRW W^ENTRY_MASK :START WITH 0
50 0000'CF47 D0 0172 366 REGLST:: :REGLIS = REGLIS MASK_ITEM
00 0002'CF 50 E3 0172 367 MOVL W^MAC$AL_VALSTACK[R7],R0 :GET THE MASK BIT NUMBER
05 0178 368 BBCS R0,W^ENTRY_MASK,10$ :SET THE BIT IN THE MASK
017E 369 10$: RSB
017F 370
017F 371 :++
017F 372 : FUNCTIONAL DESCRIPTION:
017F 373 :
017F 374 : MASK IS CALLED WHEN AN ENTRY-POINT MASK HAS BEEN ACCUMULATED
017F 375 : IN ENTRY MASK. IF WE ARE EVALUATEING EXPRESSIONS THE VALUE
017F 376 : WILL BE STACKED IN PASS 2.
017F 377 :
017F 378 :--
017F 379
017F 380 .ENABL LSB
017F 381
017F 382 MASK:: :REGISTER MASK = DUPM DANGOPN REGLIS DANGCLS
50 0002'CF 3C 017F 383 MOVZWL W^ENTRY_MASK,R0 :PICK UP MASK WORD
10 11 0184 384 BRB 10$ :FINISH IN COMMON CODE
0186 385
0186 386 :++
0186 387 : FUNCTIONAL DESCRIPTION:
0186 388 :
0186 389 : MASKX IS CALLED WHEN '^MRn' IS SCANNED. A MASK IS CREATED
0186 390 : AND THE VALUE IS SENT TO PASS 2 IF EXPRESSIONS ARE BEING
0186 391 : EVALUATED.
0186 392 :
0186 393 :--
0186 394
0186 395 MASKX:: :REGISTER_MASK = DUPM MASK_ITEM
51 0000'CF47 D0 0186 396 MOVL W^MAC$AL_VALSTACK[R7],R1 :GET MASK_BIT NUMBER
04 50 50 D4 018C 397 CLRL R0 :START WITH A CLEAN SLATE
02 11 E3 018E 398 BBCS R1,R0,10$ :SET THE MASK BIT AND JOIN COMMON CODE
0192 399 BRB 10$ :BETTER SAFE THAN SORRY
0194 400
0194 401 :++
0194 402 : FUNCTIONAL DESCRIPTION:
0194 403 :
0194 404 : MASKNL IS CALLED WHEN A NULL ENTRY-MASK IS SCANNED. IF
0194 405 : EXPRESSIONS ARE BEING EVALUATED, A ZERO IS STACKED IN
0194 406 : PASS 2.
0194 407 :
```



```

0000'CF 50 D4 0194 408 ;--
          50 D0 0194 409
          06 E1 0194 410 MASKNL::
0000'CF 59 D0 0194 411 CLRL R0 ;REGISTER MASK = DUPM DANGOPN DANGCLS
          05 D0 0196 412 10$: MOVL R0,W^MAC$GL_VALUE ;RESULT IS 0
          05 E1 019B 413 BBC #FLG$V EVALEXP, (R11), 20$ ;STORE RESULT
          05 D0 019F 414 $INTOUT_LW INT$ STKL, <W^MAC$GL_VALUE> ;BRANCH IF NO EXPRESSION EVALUATION
          05 D0 01A9 415 MOVL -R9,W^MAC$GL_EXPEND ;YES--SEND VALUE TO PASS 2
          05 D0 01AE 416 20$: RSB ;SAVE END OF EXPRESSION
          05 D0 01AF 417
          05 D0 01AF 418 .DSABL LSB

```

```
01AF 420
01AF 421
01AF 422 :++
01AF 423 :
01AF 424 :
01AF 425 :
01AF 426 :
01AF 427 :
01AF 428 :
01AF 429 :
01AF 430 :
01AF 431 :
01AF 432 :
01AF 433 :
01AF 434 :
01AF 435 :
01AF 436 :
01AF 437 :
01AF 438 :
01AF 439 :
01AF 440 :--
01AF 441 :
01AF 442 EXPBIN::
55 57 D0 01AF 443
01B2 444
01BD 445
56 FFFC'CF45 D0 01C8 446
0B 6B 06 E1 01CE 447
50 56 D0 01D2 448
FE28' 30 01D5 449
0000'CF 59 D0 01D8 450
56 D5 01DD 451 10$:
58 13 01DF 452
0A 56 91 01E1 453
10 12 01E4 454
01 0000'CF D1 01E6 455
09 15 01EB 456
05 6B 02 E1 01ED 457
0000'CF 02 C2 01F1 458
0000'CF D4 01F6 459 20$:
56 DD 01FA 460
56 0000'CF46 D0 01FC 461
66 16 0202 462
56 8ED0 0204 463
51 0000'CF D0 0207 464
33 13 020C 465
0000'CF D5 020E 466
2D 12 0212 467
52 007D8810 8F D0 0214 468
04 56 91 021B 469
0B 12 021E 470
51 D5 0220 471
07 18 0222 472
52 007D8808 8F D0 0224 473
022B 474 30$:
08 11 0237 475
0239 476 ;
```

## .SBTTL EXPRESSIONS

VAX-11 MACRO RECOGNIZES DIFFERENT TYPES OF EXPRESSIONS. THESE ROUTINES PROCESS COMPILE-TIME EXPRESSIONS. THE RESULT OF SUCH AN EXPRESSION IS A LONGWORD WHICH WILL BE KNOWN BY PASS2 (OR AT LINK TIME IF THE EXPRESSION INVOLVES GLOBALS OR EXTERNALS). THE MOST COMMON USAGE OF THIS TYPE OF EXPRESSIONS IS IN OPERANDS. ANOTHER TYPE OF EXPRESSION IS FOUND IN THE ASSIGNMENT STATEMENT WHERE AN EXPRESSION GENERATES CODE TO EVALUATE THE EXPRESSION AT RUN TIME.

THE 'PRIMITIVE' ROUTINES SET FLAGS DESCRIBING THE EXPRESSION. THESE FLAGS MUST BE INITIALIZED BY THE EXPRESSION CALLER IF THEY ARE TO BE USED.

FLG\$M\_COMPEXPR FALSE IF EXPRESSION VALUE NOT YET KNOWN  
FLG\$M\_EVALEXPR TRUE CAUSES EVALUATION TO OCCUR ON PASS 2  
FLG\$M\_ABSEXPR TRUE INDICATES THAT EXPRESSION IS ABSOLUTE

```
EXPBIN::
MOV L R7,R5 ;EXPR = EXPR OPBINARY PRIMARY
$V PUSH W^MAC$AL_VALSTACK-8[R5] ;COPY STACK POINTER
$V PUSH W^MAC$AL_VALSTACK[R5] ;PUSH LEFT OPERAND ONTO STACK
MOV L W^MAC$AL_VALSTACK-4[R5],R6 ;PUSH RIGHT OPERAND
BBC #FLG$V_EVALEXPR,(R11),10$ ;GET COMMAND FROM STACK
MOV L R6,R0 ;EVALUATE ON PASS 2?
BSBW MAC$INTOUT X ;YES--GET COMMAND
MOV L R9,W^MAC$GL_EXPEND ;OUTPUT CMD TO INT FILE
TST L R6 ;SAVE END OF EXPRESSION PTR
BEQ L 40$ ;WAS ROUTINE SUPPLIED?
CMPB R6,#INT$_SUB ;IF EQL NO
BNEQ 20$ ;SUBTRACTION?
CMPL W^MAC$GL_ABSFLAG,#1 ;IF NEQ NO
BLEQ 20$ ;YES--SEVERAL RELATIVE REFS?
BBC #FLG$V_COMPEXPR,(R11),20$ ;IF LEQ NO
SUBL2 #2,W^MAC$GL_ABSFLAG ;YES--REALLY COMPILE TIME EXPR?
CLRL W^MAC$GL_VAL3 ;YES--MAKE RESULT ABSOLUTE
PUSHL R6 ;CLEAR EXPRESSION OVERFLOW IND.
MOV L W^P1$ARITH_DISP[R6],R6 ;SAVE ROUTINE IDENT.
JSB (R6) ;GET ROUTINE ADDRESS
POPL R6 ;CALL ROUTINE
MOV L W^MAC$GL_VAL3,R1 ;RESTORE ROUTINE IDENT.
BEQ L 50$ ;EXPRESSION OVERFLOW?
TST L W^MAC$GL_ABSFLAG ;IF EQL NO
BNEQ 50$ ;YES--ABSOLUTE EXPRESSION?
MOV L #MAC$ EXPOVR32,R2 ;IF NEQ NO
CMPB R6,#INT$_DIV ;No--assume expression overflow
BNEQ 30$ ;UNLESS IT WAS DIVISION
TST L R1 ;IF NEQ NO
BGEQ 30$ ;THEN CHECK FOR DIVIDE BY 0
MOV L #MAC$ DIVBYZERO,R2 ;IF GEQ THEN NOT DIVIDE BY 0
$INTOUT_LW INT$_WRN,<R2,W^MAC$GL_ERRPT> ;It was divide by zero
BRB 50$ ;EMIT ERROR TO PASS 2
```



MAC\$ACTPRI  
V04-000

PRIMARIES  
EXPRESSIONS

B 4

16-SEP-1984 02:00:18 VAX/VMS Macro V04-00  
5-SEP-1984 01:47:04 [MACRO.SRC]ACTPRI.MAR;1

Page 12  
(8)

		0239	477	:	NO ROUTINE SUPPLIED		
		0239	478	:			
		0239	479	40\$:	\$VPUSH #0	;RESULT IS 0	
01	0000'CF	D1	0241	480	50\$:	\$VPOP W^MAC\$GL_VALUE	;POP RESULT INTO MAC\$GL_VALUE
	OC	15	024B	481		CMPL W^MAC\$GL_ABSFLAG,#1	;SEVERAL RELATIVE REFERENCES?
05	00000000'GF	E9	0250	482		BLEQ 60\$	;IF LEQ NO
	0000'CF 01	9A	0252	483		BLBC G^MAC\$GL_ABSFLAG,60\$	;YES -- ARE THE NUMBER OF REFS ODD?
		05	0259	484		MOVZBL #1,W^MAC\$GL_ABSFLAG	;YES -- CALL IT ONE (RESULT IS...
			025E	485	60\$:	RSB	;...THE EXPRESSION IS RELOCATEABLE)

```
025F 487 .SBTTL UP-ARROW-A ASCII TEXT PRIMARY
025F 488
025F 489 :++
025F 490 : FUNCTIONAL DESCRIPTION:
025F 491 :
025F 492 : NUMASC IS INVOKED WHEN THE PRODUCTION 'UP-ARROW-A' IS
025F 493 : FOUND IN THE INPUT. IT SCANS THE NEXT CHARACTER AS A
025F 494 : DELIMITER, THEN READS TEXT, STORING UP TO THE MAXIMUM
025F 495 : NUMBER OF CHARACTERS IN 'MAC$GL_VALUE', LOOKING FOR
025F 496 : THE MATCHING DELIMITER. IF THE MAXIMUM NUMBER OF BYTES
025F 497 : FOR THIS OPERAND IS EXCEEDED OR IF END-OF-LINE IS FOUND
025F 498 : BEFORE THE MATCHING DELIMITER, A MESSAGE IS OUTPUT
025F 499 : TO PASS 2.
025F 500 :
025F 501 :--
025F 502
025F 503 NUMASC::
00 6B 24 E3 025F 504 BBSC #FLG$V UPAFLG,(R11),.+1 ; SPECIAL OPERATOR = DUPA
FD9A' 30 0263 505 BSBW MAC$SKIPSP ; FLAG DUPA WAS SEEN
56 0000'CF 9E 0266 506 MOVAB W^MAC$GQ_VALUEQ,R6 ; SKIP SPACES AND TABS
66 7C 026B 507 CLRQ (R6) ; POINT TO RESULT AREA
08 A6 7C 026D 508 CLRQ 8(R6) ; CLEAR OUT 8 BYTES
55 5A D0 0270 509 MOVL R10,R5 ; and then the next 8 bytes
0D 55 91 0273 510 CMPB R5,#CR ; COPY DELIMITER
1F 13 0276 511 BEQL 20$ ; IS DELIMITER CR?
54 0000'CF 9A 0278 512 MOVZBL W^MAC$GL_OPSIZE,R4 ; IF EQL YES--ERROR
00 6B 2F E3 027D 513 BBSC #FLG$V DLIMSTR,(R11),.+1 ; GET MAX SIZE OF OPERAND
FD7C' 30 0281 514 10$: BSBW MAC$GETCHR ; PASS ALL CHARACTERS (EVEN -;)
55 5A 91 0284 515 CMPB R10,R5 ; GET NEXT CHARACTER
1C 13 0287 516 BEQL 30$ ; DELIMITER?
0D 5A 91 0289 517 CMPB R10,#CR ; IF EQL YES
09 13 028C 518 BEQL 20$ ; NO--END OF LINE?
54 D7 028E 519 DECL R4 ; IF EQL YES--ERROR
EF 19 0290 520 BLSS 10$ ; NO--ROOM TO STORE BYTE?
86 5A 90 0292 521 MOVB R10,(R6)+ ; DON'T STORE IF TOO MANY CHARS
EA 11 0295 522 BRB 10$ ; STORE CHARACTER
0297 523 : ; LOOP FOR MORE
0297 524 : FOUND EOL BEFORE DELIMITER
0297 525 :
0297 526 20$: $MAC_ERR UNTERMARG ; Get message code
09 6B FD61' 30 029C 527 BSBW MAC$ERRORPT ; ISSUE MESSAGE TO PASS 2
2F E4 029F 528 BBSC #FLG$V_DLIMSTR,(R11),40$ ; CLEAR ALLCHR AND GO FINISH UP
07 11 02A3 529 BRB 40$ ; FINISH
02A5 530 :
02A5 531 : FOUND OTHER DELIMITER
02A5 532 :
00 6B 2F E4 02A5 533 30$: BBSC #FLG$V_DLIMSTR,(R11),.+1 ; DO NOT PASS ALL CHARACTERS
FD54' 30 02A9 534 BSBW MAC$GETCHR ; SKIP OVER DELIMITER
54 D5 02AC 535 40$: TSTL R4 ; TOO MANY CHARACTERS?
10 18 02AE 536 BGEQ 50$ ; IF GEQ NO
02B0 537 $INTOUT_LW INT$ WRN,<#MAC$_DATATRUNC,W^MAC$GL_ERRPT> ; Yes--report error
02C0 538 50$: CMPB W^MAC$GL_OPSIZE,#8 ; Was this a QUAD or OCTA operand?
08 0000'CF 91 02C0 539 BLSS 70$ ; No if LSS
0000'CF 0000'CF D0 02C5 540 MOVL W^MAC$GL_VAL3,- ; Yes: save bits 32 to 63
02C7 541 W^MAC$GL_HIGH 32
02CE 542 CMPB W^MAC$GL_OPSIZE,#16 ; Was this an OCTA operand?
10 0000'CF 91 02CE 543
```



MAC\$ACTPRI  
V04-000

PRIMARIES  
UP-ARROW-A ASCII TEXT PRIMARY

D 4

16-SEP-1984 02:00:18 VAX/VMS Macro V04-00  
5-SEP-1984 01:47:04 [MACRO.SRC]ACTPRI.MAR;1

Page 14  
(9)

0000'CF	07	12	02D3	544	BNEQ	70\$	; No if NEQ
0000'CF	7D	02D5	545	MOVQ	W^MAC\$GQ_VAL2, -		; Yes: save bits 64 to 127
		02DC	546		W^MAC\$GQ_HIGH_64		
FE29	31	02DC	547	70\$: BRW	PRMINT		;TREAT AS INTEGER DATA

```
02DF 549 .SBTTL RADIX CONTROL
02DF 550
02DF 551 :++
02DF 552 : FUNCTIONAL DESCRIPTION:
02DF 553 :
02DF 554 : THESE FOUR ROUTINES ARE INVOKED WHEN A RADIX CONTROL
02DF 555 : PRIMARY IS ENCOUNTERED. THESE ROUTINES SAVE THE
02DF 556 : OLD RADIX IN MAC$GL VALUE (FOR LATER RESTORATION) AND
02DF 557 : SET THE NEW RADIX IN MAC$GB_RDXNDX.
02DF 558 :
02DF 559 :--
02DF 560
02DF 561 RDXBIN:: ;RADIX_CONTROL = DUPB
0A 10 02DF 562 BSBB SET_RADIX ;GO SET THE INDEX FOR BINARY
00 00 02E1 563 .BYTE RDX$V_BINARY
02E2 564
02E2 565 RDXDEC:: ;RADIX_CONTROL = DUPD
07 10 02E2 566 BSBB SET_RADIX ;SET DECIMAL RADIX
02 02 02E4 567 .BYTE RDX$V_DECIMAL
02E5 568
02E5 569 RDXOCT:: ;RADIX_CONTROL = DUPO
04 10 02E5 570 BSBB SET_RADIX ;SET OCTAL RADIX
01 01 02E7 571 .BYTE RDX$V_OCTAL
02E8 572
02E8 573 RDXHEX:: ;RADIX_CONTROL = DUPX
01 10 02E8 574 BSBB SET_RADIX ;SET HEX RADIX
03 03 02EA 575 .BYTE RDX$V_HEX
02EB 576
02EB 577 SET_RADIX:
02EB 578
0000'CF 9A 02EB 579 MOVZBL W^MAC$GB_RDXNDX,- ;SAVE CURRENT INDEX
0000'CF 02EF 580 W^MAC$GL_VALUE
0000'CF 9E 90 02F2 581 MOV B @ (SP)+,W^MAC$GB_RDXNDX ;SET NEW RADIX AND CLEAN STACK
05 02F7 582 RSB ;RETURN TO CALLER'S CALLER
02F8 583
```



```
02F8 585 .SBTTL OPERATORS
02F8 586
02F8 587 :++
02F8 588 : FUNCTIONAL DESCRIPTION:
02F8 589 :
02F8 590 : THESE OPERATOR ROUTINES ARE CALLED WHEN A BINARY OPERATOR
02F8 591 : IS ENCOUNTERED IN THE TEXT. THESE ROUTINES MERELY SET
02F8 592 : THE OPERATOR NUMBER INTO MAC$GL VALUE FOR LATER PROCESSING
02F8 593 : BY THE EXPRESSION EVALUATION ROUTINE (EXPBIN).
02F8 594 :
02F8 595 :--
02F8 596
02F8 597
02F8 598
02F8 599 .MACRO OP OPR
02F8 600 BSBB SET_UP_OPERATOR
02F8 601 .BYTE INT$_OPR
02F8 602 .ENDM
02F8 603
02F8 604
02F8 605 OPPLUS:: ;OPBINARY = DDPLUS
02F8 606 OP OP ADD
02F8 607 OPMINU:: ;OPBINARY = DDMINUS
02F8 608 OP SUB
02FE 609 OPMUL:: ;OPBINARY = DDTIMES
02FE 610 OP MUL
0301 611 OPDIV:: ;OPBINARY = DDDIV
0301 612 OP DIV
0304 613 OPAND:: ;OPBINARY = DDAND
0304 614 OP AND
0307 615 OPOR:: ;OPBINARY = DDOR
0307 616 OP OR
030A 617 OPXOR:: ;OPBINARY = DDXOR
030A 618 OP XOR
030D 619 OPASH:: ;OPBINARY = DDASH
030D 620 OP ASH
0310 621 OPCOM:: ;OPBINARY = DDUPC
0310 622 OP NOT
0313 623 OPNEG:: ;OPBINARY = DDMINUS
0313 624 OP NEG
0316 625 OPSAME:: ;OPBINARY = DDPLUS
0316 626 OP SAME
0319 627
0319 628 SET_UP_OPERATOR:
0000'CF 9E 9A 0319 629 MOVZBL @ (SP)+,W^MAC$GL_VALUE ;GET THE OPERATOR NUMBER
05 031E 630 RSB ;RETURN TO CALLER'S CALLER
```

```
.SBTTL SYMBOL ATTRIBUTE DIRECTIVES -GLOBL/DEBUG/WEAK/EXTRN

031F 632
031F 633
031F 634 :++
031F 635 : FUNCTIONAL DESCRIPTION:
031F 636 :
031F 637 : GLOBAL/DEBUG/WEAK/EXTRN ARE CALLED WHEN THE CORRESPONDING
031F 638 : DIRECTIVE IS SCANNED. FLAGS ARE SET IN SYM_FLAG FOR THE
031F 639 : ROUTINE 'IDLIST'. 'IDLIST' IS CALLED FOR EACH SYMBOL IN
031F 640 : THE LIST AND IT SETS THE BITS IN SYM_FLAG IN THE SYMBOL
031F 641 : BLOCK FOR THAT SYMBOL.
031F 642 :
031F 643 :--
031F 644
031F 645 GLOBAL::                                ;ID LIST HEAD = KGLOBL
OE 10 031F 646 BSBB SET SYM_FLAG ;SET FLAG TO REMEMBER
0004 0321 647 .WORD SYMSM_GLOBL
0323 648
0323 649 DEBUG::                                ;ID LIST HEAD = KDEBUG
OA 10 0323 650 BSBB SET SYM_FLAG ;SET FLAGS TO REMEMBER
00A0 0325 651 .WORD SYMSM_DEBUG!SYMSM_REF
0327 652
0327 653 WEAK::                                ;ID LIST HEAD = KWEAK
06 10 0327 654 BSBB SET SYM_FLAG ;SET FLAGS TO REMEMBER
0006 0329 655 .WORD SYMSM_WEAK!SYMSM_GLOBL
032B 656
032B 657 EXTRN::                                ;ID LIST HEAD = KEXTRN
02 10 032B 658 BSBB SET SYM_FLAG ;SET THE FLAG
0008 032D 659 .WORD SYMSM_EXTRN
032F 660
032F 661
032F 662 SET_SYM_FLAG:
0000'CF 9E B0 032F 663 MOVW @ (SP)+,W^SYM_FLAG ;REMEMBER THE FLAG BIT
05 0334 664 RSB ;RETURN TO CALLER'S CALLER
0335 665
0335 666 :++
0335 667 : FUNCTIONAL DESCRIPTION:
0335 668 :
0335 669 : AFTER A GLOBAL/DEBUG/WEAK/EXTRN DIRECTIVE HAS BEEN SCANNED,
0335 670 : 'IDLIST' IS CALLED FOR EACH SYMBOL IN THE LIST OF SYMBOLS
0335 671 : ACCOMPANYING THE DIRECTIVE. THE FLAGS CONTAINED IN SYM_FLAG
0335 672 : ARE SET FOR THE SYMBOL. IF THE DIRECTIVE IS .EXTRN AND
0335 673 : THE SYMBOL IS ALREADY DEFINED, AN ERROR MESSAGE IS ISSUED
0335 674 : TO PASS 2.
0335 675 :
0335 676 :--
0335 677
0335 678 IDLIST::                                ;ID LIST = ID
56 0000'CF47 D0 0335 679 MOVL W^MAC$AL VALSTACK[R7],R6 ;GET POINTER TO SYMBOL BLOCK
OD 0000'CF 03 E1 033B 680 BBC #SYMSV_EXTRN,W^SYM_FLAG,10$ ;BRANCH IF NOT .EXTRN
08 09 A6 00 E1 0341 681 BBC #SYMSV_DEF,SYMSW_FLAG(R6),10$ ;BRANCH IF SYMBOL NOT DEFINED
0346 682 $MAC_ERR SYMDEFINMO ; Yes--get error message
FCB2' 30 034B 683 BSBW MAC$ERRORPT ;SYMBOL DECLARED EXTERNAL BUT ALREADY DEFINE
09 A6 0000'CF A8 034E 684 10$: BISW W^SYM_FLAG,SYMSW_FLAG(R6) ;SET BIT(S) IN SYMBOL FLAGS
05 09 A6 05 E1 0354 685 BBC #SYMSV_DEBUG,SYMSW_FLAG(R6),20$ ;BRANCH IF NOT .DEBUG
00 09 A6 0E E4 0359 686 BBSC #SYMSV_SUPR,SYMSW_FLAG(R6),+1 ;DEBUG--CLEAR SUPR BIT
55 00'8F 9A 035E 687 20$: MOVZBL #CRFSK_REF,R5 ;SET REFERENCE
50 00000000'EF D0 0362 688 MOVL MAC$GL_PSECTPTR,R0 ;GET POINTER TO PSECT DATA
```



MAC\$ACTPRI  
V04-000

PRIMARIES

SYMBOL ATTRIBUTE DIRECTIVES -GLOBL/DEBUG

H 4

16-SEP-1984 02:00:18

5-SEP-1984 01:47:04

VAX/VMS Macro V04-00

[MACRO.SRC]ACTPRI.MAR;1

Page 18  
(12)

```
06 0D A0 03 E1 0369 689 BBC #PSC$V REL, - ;IF ABS PSECT
09 A6 0800 8F A8 036E 690 PSC$W OPTIONS(R0),30$ ;THEN SKIP
FC89' 31 036E 691 #SYMS$ RELPSECT,SYMS$W_FLAG(R6) ;SET REL PSECT FLAG
0374 692 30$: BRW MAC$CREF_SYM ;CREF SYMBOL IF CREFFING AND RETURN
0377 693
0377 694 .END
```

MAC\$ACTPRI  
Symbol table

PRIMARIES

1 4

16-SEP-1984 02:00:18  
5-SEP-1984 01:47:04

VAX/VMS Macro V04-00  
[MACRO.SRC]ACTPRI.MAR;1

Page 19  
(12)

\$COUNT = 0000003B  
ARG\$K\_SIZE = 000003E8  
AUD\$K\_SIZE = 00000010  
BLNK = 00000020  
CHRS\$M\_COMMA CR = 00000020  
CHRS\$M\_ILL\_CHR = 00000040  
CHRS\$M\_NUM\_BER = 00000010  
CHRS\$M\_SPA\_MSK = 00000001  
CHRS\$M\_SYM\_CH1 = 00000008  
CHRS\$M\_SYM\_CHR = 00000004  
CHRS\$M\_SYM\_DLM = 00000002  
CHRS\$V\_COMMA CR = 00000005  
CHRS\$V\_CVTLWC = 00000061  
CHRS\$V\_ILL\_CHR = 00000006  
CHRS\$V\_NOCVT = 0000007F  
CHRS\$V\_NUM\_BER = 00000004  
CHRS\$V\_SPA\_MSK = 00000000  
CHRS\$V\_SYM\_CH1 = 00000003  
CHRS\$V\_SYM\_CHR = 00000002  
CHRS\$V\_SYM\_DLM = 00000001  
CNT = 00000002  
CR = 0000000D  
CRF\$K\_REF = \*\*\*\*\* X 05  
DEBUG = 00000323 RG 05  
ENTRY\_MASK = 00000002 R 04  
ERR = 00000001  
EXPBIN = 000001AF RG 05  
EXTRN = 0000032B RG 05  
FF = 0000000C  
FLG\$M\_ALLCHR = 00000001  
FLG\$M\_BOL = 00000002  
FLG\$M\_CHKLPND = 00100000  
FLG\$M\_COMPEXPR = 00000004  
FLG\$M\_CONT = 00000008  
FLG\$M\_CRF = 40000000  
FLG\$M\_CRSEEN = 00000001  
FLG\$M\_DATRPT = 00000010  
FLG\$M\_DBGOUT = 00004000  
FLG\$M\_DLMSTR = 00008000  
FLG\$M\_ENDMCH = 00000020  
FLG\$M\_EVALEXPR = 00000040  
FLG\$M\_EXPOPT = 00000080  
FLG\$M\_EXTERR = 00010000  
FLG\$M\_EXTWRN = 00020000  
FLG\$M\_FIRSTLN = 00000200  
FLG\$M\_IFSTAT = 00800000  
FLG\$M\_IIF = 00400000  
FLG\$M\_INSERT = 00000100  
FLG\$M\_IRPC = 20000000  
FLG\$M\_LEXOP = 00000002  
FLG\$M\_LSTXST = 00000200  
FLG\$M\_MAC2COL = 00000800  
FLG\$M\_MACL = 00000800  
FLG\$M\_MACLTB = 08000000  
FLG\$M\_MACTXT = 00010000  
FLG\$M\_MEBLST = 00001000  
FLG\$M\_MOREARG = 00002000

FLG\$M\_MOREINP = 00000008  
FLG\$M\_NEWPND = 00000400  
FLG\$M\_NOREF = 01000000  
FLG\$M\_NTTYPEPC = 00000020  
FLG\$M\_NULCHR = 00040000  
FLG\$M\_OBJXST = 00200000  
FLG\$M\_OPNDCHK = 00000100  
FLG\$M\_OPRND = 00002000  
FLG\$M\_OPTVFLIDX = 00001000  
FLG\$M\_ORDLST = 00020000  
FLG\$M\_P2 = 00004000  
FLG\$M\_RPTIRP = 10000000  
FLG\$M\_SEQFIL = 02000000  
FLG\$M\_SKAN = 00008000  
FLG\$M\_SPECOP = 00000004  
FLG\$M\_SPLALL = 04000000  
FLG\$M\_STOIMF = 00040000  
FLG\$M\_SYM2COL = 00000400  
FLG\$M\_TOCLG = 00080000  
FLG\$M\_UPDFIL = 00000010  
FLG\$M\_UPMARG = 00000080  
FLG\$M\_XCRF = 00000040  
FLG\$M\_XCRF = 80000000  
FLG\$V\_ALLCHR = 00000000  
FLG\$V\_BOL = 00000001  
FLG\$V\_CHKLPND = 00000014  
FLG\$V\_COMPEXPR = 00000002  
FLG\$V\_CONT = 00000003  
FLG\$V\_CRF = 0000001E  
FLG\$V\_CRSEEN = 00000020  
FLG\$V\_DATRPT = 00000004  
FLG\$V\_DBGOUT = 0000002E  
FLG\$V\_DLMSTR = 0000002F  
FLG\$V\_ENDMCH = 00000005  
FLG\$V\_EVALEXPR = 00000006  
FLG\$V\_EXPOPT = 00000007  
FLG\$V\_EXTERR = 00000030  
FLG\$V\_EXTWRN = 00000031  
FLG\$V\_FIRSTLN = 00000029  
FLG\$V\_IFSTAT = 00000017  
FLG\$V\_IIF = 00000016  
FLG\$V\_INSERT = 00000008  
FLG\$V\_IRPC = 0000001D  
FLG\$V\_LEXOP = 00000021  
FLG\$V\_LSTXST = 00000009  
FLG\$V\_MAC2COL = 0000002B  
FLG\$V\_MACL = 0000000B  
FLG\$V\_MACLTB = 0000001B  
FLG\$V\_MACTXT = 00000010  
FLG\$V\_MEBLST = 0000000C  
FLG\$V\_MOREARG = 0000002D  
FLG\$V\_MOREINP = 00000023  
FLG\$V\_NEWPND = 0000000A  
FLG\$V\_NOREF = 00000018  
FLG\$V\_NTTYPEPC = 00000025  
FLG\$V\_NULCHR = 00000032  
FLG\$V\_OBJXST = 00000015

FLG\$V\_OPNDCHK = 00000028  
FLG\$V\_OPRND = 0000000D  
FLG\$V\_OPTVFLIDX = 0000002C  
FLG\$V\_ORDLST = 00000011  
FLG\$V\_P2 = 0000000E  
FLG\$V\_RPTIRP = 0000001C  
FLG\$V\_SEQFIL = 00000019  
FLG\$V\_SKAN = 0000000F  
FLG\$V\_SPECOP = 00000022  
FLG\$V\_SPLALL = 0000001A  
FLG\$V\_STOIMF = 00000012  
FLG\$V\_SYM2COL = 0000002A  
FLG\$V\_TOCLG = 00000013  
FLG\$V\_UPDFIL = 00000024  
FLG\$V\_UPMARG = 00000027  
FLG\$V\_XCRF = 0000001F  
GLOBAL = 0000031F RG 05  
HASHSZ = 0000007F  
HYPHEN = 0000002D  
IDLST = 00000335 RG 05  
INP\$K\_BUFSIZ = 000003E8  
INT\$K\_BUFSIZ = 000013F4  
INT\$K\_BUFWRN = 00001390  
INT\$\_ADD = 00000001  
INT\$\_AND = 00000002  
INT\$\_ASH = 00000003  
INT\$\_ASN = 0000000C  
INT\$\_AUGPC = 0000000D  
INT\$\_BDST = 0000000E  
INT\$\_CHKL = 0000000F  
INT\$\_DIV = 00000004  
INT\$\_END = 00000010  
INT\$\_EPT = 00000011  
INT\$\_ERR = 00000012  
INT\$\_ETX = 00000013  
INT\$\_FNEWL = 00000014  
INT\$\_ILG = 00000000  
INT\$\_INFO = 0000003A  
INT\$\_LGLAB = 00000015  
INT\$\_MACL = 00000016  
INT\$\_MUL = 00000005  
INT\$\_NEG = 00000006  
INT\$\_NEWL = 00000017  
INT\$\_NEWP = 00000018  
INT\$\_NOT = 00000007  
INT\$\_OP = 00000019  
INT\$\_OR = 00000008  
INT\$\_PRIL = 0000001A  
INT\$\_PRT = 0000001B  
INT\$\_PSECT = 0000001C  
INT\$\_REDEF = 0000001D  
INT\$\_REF = 0000001E  
INT\$\_REST = 0000001F  
INT\$\_SAME = 00000009  
INT\$\_SAVE = 00000020  
INT\$\_SBTTL = 00000021

MA  
VO



MAC\$ACTPRI  
Symbol table

## PRIMARIES

J 4

16-SEP-1984 02:00:18  
5-SEP-1984 01:47:04VAX/VMS Macro V04-00  
[MACRO.SRC]ACTPRI.MAR;1Page 20  
(12)

INT\$ SETFLAG = 00000022  
INT\$ SETLONG = 00000023  
INT\$ SPIC = 00000024  
INT\$ SPID = 00000025  
INT\$ STIB = 00000026  
INT\$ STIL = 00000028  
INT\$ STIW = 00000027  
INT\$ STKEPT = 00000029  
INT\$ STKG = 0000002A  
INT\$ STKL = 0000002B  
INT\$ STKPC = 0000002C  
INT\$ STKS = 0000002D  
INT\$ STOB = 00000034  
INT\$ STOL = 0000002E  
INT\$ STOW = 00000035  
INT\$ STRB = 0000002F  
INT\$ STRL = 00000031  
INT\$ STRSB = 00000032  
INT\$ STRSW = 00000033  
INT\$ STRW = 00000030  
INT\$ STSB = 00000036  
INT\$ STSW = 00000037  
INT\$ SUB = 0000000A  
INT\$ SUME = 00000039  
INT\$ WRN = 00000038  
INT\$ XOR = 0000000B  
LST\$K\_BUF\$SIZ = 00000086  
LST\$K\_L\_P\_PAGE = 0000003C  
LST\$K\_TITLE\_SIZ = 00000028  
MAC\$AC\_VALSTACK \*\*\*\*\* X 05  
MAC\$CREF\_SYM \*\*\*\*\* X 05  
MAC\$ERRORPT \*\*\*\*\* X 05  
MAC\$GB\_RDXNDX \*\*\*\*\* X 05  
MAC\$GETCHR \*\*\*\*\* X 05  
MAC\$GETFLOAT \*\*\*\*\* X 05  
MAC\$GL\_ABSFLAG \*\*\*\*\* X 05  
MAC\$GL\_ERRPT \*\*\*\*\* X 05  
MAC\$GL\_EXPEND \*\*\*\*\* X 05  
MAC\$GL\_HIGH\_32 \*\*\*\*\* X 05  
MAC\$GL\_OP\$SIZE \*\*\*\*\* X 05  
MAC\$GL\_PC \*\*\*\*\* X 05  
MAC\$GL\_P\$MSEG \*\*\*\*\* X 05  
MAC\$GL\_P\$SECT \*\*\*\*\* X 05  
MAC\$GL\_P\$SECTPTR \*\*\*\*\* X 05  
MAC\$GL\_VAL3 \*\*\*\*\* X 05  
MAC\$GL\_VALUE \*\*\*\*\* X 05  
MAC\$GL\_HIGH\_64 \*\*\*\*\* X 05  
MAC\$GL\_VAL2 \*\*\*\*\* X 05  
MAC\$GL\_VALUEQ \*\*\*\*\* X 05  
MAC\$INTERR\_2\_LW \*\*\*\*\* X 05  
MAC\$INTOUT\_1\_LW \*\*\*\*\* X 05  
MAC\$INTOUT\_X \*\*\*\*\* X 05  
MAC\$SKIPSP \*\*\*\*\* X 05  
MAC\$DATATRUNC = 007D8800  
MAC\$DIVBYZERO = 007D8808  
MAC\$EXPOVR32 = 007D8810  
MAC\$SYMDEFINMO = 007D91E2

MAC\$\_INTERMARG = 007D922A  
MAC\$\_SUBSYS = 0000007D  
MASK = 0000017F RG 05  
MASKNL = 00000194 RG 05  
MASKX = 00000186 RG 05  
NUMASC = 0000025F RG 05  
NUMFLT = 00000100 RG 05  
OBJ\$K\_BUF\$SIZ = 00000200  
OPAND = 00000304 RG 05  
OPASH = 0000030D RG 05  
OPCOM = 00000310 RG 05  
OPDIV = 00000301 RG 05  
OPF\$M\_LASTOPR = 00002000  
OPF\$M\_OPTEXP = 00001000  
OPF\$V\_LASTOPR = 0000000D  
OPF\$V\_OPTEXP = 0000000C  
OPMINO = 000002FB RG 05  
OPMUL = 000002FE RG 05  
OPNEG = 00000313 RG 05  
OPOR = 00000307 RG 05  
OPPLUS = 000002F8 RG 05  
OPSAME = 00000316 RG 05  
OPXOR = 0000030A RG 05  
P1\$ARITH\_ADD \*\*\*\*\* X 03  
P1\$ARITH\_AND \*\*\*\*\* X 03  
P1\$ARITH\_ASH \*\*\*\*\* X 03  
P1\$ARITH\_DISP = 00000000 RG 03  
P1\$ARITH\_DIV \*\*\*\*\* X 03  
P1\$ARITH\_MUL \*\*\*\*\* X 03  
P1\$ARITH\_NEG \*\*\*\*\* X 03  
P1\$ARITH\_NOT \*\*\*\*\* X 03  
P1\$ARITH\_OR \*\*\*\*\* X 03  
P1\$ARITH\_SAME \*\*\*\*\* X 03  
P1\$ARITH\_SUB \*\*\*\*\* X 03  
P1\$ARITH\_XOR \*\*\*\*\* X 03  
PRMBRK = 0000011C RG 05  
PRMINT = 00000108 RG 05  
PRMPC = 0000012E RG 05  
PRMRDX = 00000125 RG 05  
PRMSYM = 00000063 RG 05  
PRMUN = 00000000 RG 05  
PSC\$B\_NAME = 00000004  
PSC\$B\_SEG = 0000000C  
PSC\$B\_UNUSED = 0000000B  
PSC\$K\_BLK\$SIZ = 00000013  
PSC\$K\_NO\_OPTNS = 0000000A  
PSC\$L\_CURLOC = 0000000F  
PSC\$L\_LINK = 00000000  
PSC\$L\_MAXLGTH = 00000005  
PSC\$M\_ABS = FFFFFFFF7  
PSC\$M\_ALIGNFLG = 00004000  
PSC\$M\_ALLOPTNS = 000003FF  
PSC\$M\_BYTE = 00004000  
PSC\$M\_CON = FFFFFFFFB  
PSC\$M\_DEFAULT = 000001C8  
PSC\$M\_EXE = 000000C0  
PSC\$M\_GBL = 00000010

PSC\$M\_LCL = FFFFFFFEF  
PSC\$M\_LIB = 00000002  
PSC\$M\_LONG = 00004800  
PSC\$M\_NOEXE = FFFFFFFBF  
PSC\$M\_NOPIC = FFFFFFFFE  
PSC\$M\_NORD = FFFFFFF7F  
PSC\$M\_NOSHR = FFFFFFFDF  
PSC\$M\_NOVEC = FFFFFFFDF  
PSC\$M\_NOWRT = FFFFFFFEF  
PSC\$M\_OVR = 00000004  
PSC\$M\_PAGE = 00006400  
PSC\$M\_PIC = 00000001  
PSC\$M\_QUAD = 00004C00  
PSC\$M\_RD = 00000080  
PSC\$M\_REL = 00000008  
PSC\$M\_SHR = 00000020  
PSC\$M\_USR = FFFFFFFFD  
PSC\$M\_VEC = 00000200  
PSC\$M\_WORD = 00004400  
PSC\$M\_WRT = 00000180  
PSC\$S\_ALIGNMENT = 00000004  
PSC\$V\_ALIGNFLG = 0000000E  
PSC\$V\_ALIGNMENT = 0000000A  
PSC\$V\_EXE = 00000006  
PSC\$V\_GBL = 00000004  
PSC\$V\_LIB = 00000001  
PSC\$V\_OVR = 00000002  
PSC\$V\_PIC = 00000000  
PSC\$V\_RD = 00000007  
PSC\$V\_REL = 00000003  
PSC\$V\_SHR = 00000005  
PSC\$V\_VEC = 00000009  
PSC\$V\_WRT = 00000008  
PSC\$W\_FLAG = 00000009  
PSC\$W\_OPTIONS = 0000000D  
RDX\$V\_BINARY = 00000000  
RDX\$V\_DECIMAL = 00000002  
RDX\$V\_DOUBLE = 00000005  
RDX\$V\_FLOAT = 00000004  
RDX\$V\_GFLOAT = 00000006  
RDX\$V\_HEX = 00000003  
RDX\$V\_HFLOAT = 00000007  
RDX\$V\_OCTAL = 00000001  
RDXBIN = 000002DF RG 05  
RDXDEC = 000002E2 RG 05  
RDXHEX = 000002E8 RG 05  
RDXOCT = 000002E5 RG 05  
REG\$ PC = 0000000F  
REGLST = 00000172 RG 05  
RGLST1 = 0000016E RG 05  
SEMI = 0000003B  
SET\_RADIX = 000002EB R 05  
SET\_SYM\_FLAG = 0000032F R 05  
SET\_UP\_OPERATOR = 00000319 R 05  
SIGN\_BIT = 80000000  
STB\$K\_PG\_MISS = 0000000A  
SYM\$B\_NAME = 00000004

SYMSB\_SEG 0000000C  
SYMSB\_TOKEN 0000000B  
SYMSK\_BLKSI2 0000000D  
SYMSK\_MAXLEN = 0000001F  
SYMSK\_TWOCOL = 00000010  
SYMSL\_LINK 00000000  
SYMSL\_VAL 00000005  
SYMSM\_ABS = 00000010  
SYMSM\_ASN = 00000100  
SYMSM\_CRFO = 00002000  
SYMSM\_DEBUG = 00000020  
SYMSM\_DEF = 00000001  
SYMSM\_DELMAC = 00000200  
SYMSM\_EPT = 00000200  
SYMSM\_EXTRN = 00000008  
SYMSM\_GLOBL = 00000004  
SYMSM\_LOCAL = 00000040  
SYMSM\_ODBG = 00000400  
SYMSM\_REF = 00000080  
SYMSM\_RELPSECT = 00000800  
SYMSM\_SUPR = 00004000  
SYMSM\_WEAK = 00000002  
SYMSM\_XCRF = 00001000  
SYMSV\_ABS = 00000004  
SYMSV\_ASN = 00000008  
SYMSV\_CRFO = 0000000D  
SYMSV\_DEBUG = 00000005  
SYMSV\_DEF = 00000000  
SYMSV\_DELMAC = 00000009  
SYMSV\_EPT = 00000009  
SYMSV\_EXTRN = 00000003  
SYMSV\_GLOBL = 00000002  
SYMSV\_LOCAL = 00000006  
SYMSV\_ODBG = 0000000A  
SYMSV\_REF = 00000007  
SYMSV\_RELPSECT = 0000000B  
SYMSV\_SUPR = 0000000E  
SYMSV\_WEAK = 00000001  
SYMSV\_XCRF = 0000000C  
SYMSW\_FLAG 00000009  
SYM\_FLAG 00000000 R 04  
TAB = 00000009  
WEAK 00000327 RG 05  
X1 = 00000033  
X2 = 00080000

+-----+  
! Psect synopsis !  
+-----+

PSECT name	Allocation	PSECT No.	Attributes
. ABS .	00000000 ( 0.)	00 ( 0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
. BLANK .	00000000 ( 0.)	01 ( 1.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC BYTE
\$ABSS	00000013 ( 19.)	02 ( 2.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
MAC\$RO_DATA	00000030 ( 48.)	03 ( 3.)	NOPIC USR CON REL GBL NOSHR NOEXE RD NOWRT NOVEC LONG
MAC\$ACTPRI_DATA	00000004 ( 4.)	04 ( 4.)	NOPIC USR CON REL LCL NOSHR NOEXE RD WRT NOVEC LONG



MAC\$ACTPRI  
Psect synopsis

PRIMARIES

L 4

16-SEP-1984 02:00:18 VAX/VMS Macro V04-00  
5-SEP-1984 01:47:04 [MACRO.SRC]ACTPRI.MAR;1

Page 22  
(12)

MAC\$RO\_CODE\_P1

00000377 ( 887.) 05 ( 5.) NOPIC USR CON REL GBL NOSHR EXE RD NOWRT NOVEC LONG

-----  
! Performance indicators !  
-----

Phase	Page faults	CPU Time	Elapsed Time
-----	-----	-----	-----
Initialization	35	00:00:00.03	00:00:02.43
Command processing	131	00:00:00.37	00:00:03.59
Pass 1	216	00:00:03.53	00:00:15.39
Symbol table sort	0	00:00:00.43	00:00:00.97
Pass 2	135	00:00:01.18	00:00:02.71
Symbol table output	32	00:00:00.15	00:00:00.32
Psect synopsis output	2	00:00:00.02	00:00:00.02
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	553	00:00:05.72	00:00:25.43

The working set limit was 1500 pages.

34438 bytes (68 pages) of virtual memory were used to buffer the intermediate code.

There were 30 pages of symbol table space allocated to hold 462 non-local and 33 local symbols.

694 source lines were read in Pass 1, producing 25 object records in Pass 2.

16 pages of virtual memory were used to define 15 macros.

-----  
! Macro library statistics !  
-----

Macro library name	Macros defined
-----	-----
-\$255\$DUA28:[MACRO.OBJ]MACRO.MLB;1	12
-\$255\$DUA28:[SYSLIB]STARLET.MLB;2	3
TOTALS (all libraries)	15

506 GETS were required to define 15 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS\$:ACTPRI/OBJ=OBJ\$:ACTPRI MSRC\$:ACTPRI/UPDATE=(ENH\$:ACTPRI)+LIB\$:MACRO/LIB



0224 AH-BT13A-SE  
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION  
CONFIDENTIAL AND PROPRIETARY

